

..... 3

-

Last  
update:  
2025/12/24 01:19 - [https://atl.kr/dokuwiki/doku.php/%ED%8F%B0%ED%8A%B8\\_%EC%9D%B4%EB%A6%84%EC%9C%BC%EB%A1%9C\\_%ED%8C%8C%EC%9D%BC%EB%AA%85\\_%EB%B3%80%EA%B2%BD](https://atl.kr/dokuwiki/doku.php/%ED%8F%B0%ED%8A%B8_%EC%9D%B4%EB%A6%84%EC%9C%BC%EB%A1%9C_%ED%8C%8C%EC%9D%BC%EB%AA%85_%EB%B3%80%EA%B2%BD)

-

---

— 2025/12/24 01:17

```
가 fc-scan 가 . fc-scan fontconfig
.(RHEL/Rocky )
```

```
#!/bin/bash

# ( )
export LC_ALL=C.UTF-8
export LANG=C.UTF-8

# : ( )
FONTDIR="./fonts/"

#
RED='\033[0;31m'
GREEN='\033[0;32m'
YELLOW='\033[1;33m'
BLUE='\033[0;34m'
PURPLE='\033[0;35m'
BOLD='\033[1m'
NC='\033[0m'

# ===
function show_help() {
    echo -e
    "${BLUE}===== ${NC}"
    echo -e "${GREEN}${BOLD} ${NC}"
    echo -e
    "${BLUE}===== ${NC}"
    echo -e " (Fullname) ."
    echo ""
    echo -e "${YELLOW} : ${NC}"
    echo -e " $0 ${BOLD}[verify | rename]${NC}"
    echo ""
    echo -e "${YELLOW} : ${NC}"
    echo -e " ${BOLD}verify${NC} : ${GREEN} ${NC} (
    .)"
    echo -e " ${BOLD}rename${NC} : ${RED} ${NC} (
    .)"
    echo ""
    echo -e "${YELLOW} : ${NC}"
    echo -e " ${BOLD}-h, --help${NC} : ."
    echo ""
    echo -e "${YELLOW} : ${NC}"
    echo -e " - ."
```

```
echo -e " - ."
```

```
echo -e " - 32 'VF' ."
```

```
echo -e " - 'VF'가 ."
```

```
echo -e
```

```
"${BLUE}===== ${NC}"
```

```
}
```

```
# ===
```

```
if [[ -z "$1" ]]; then
```

```
    show_help
```

```
    exit 0
```

```
fi
```

```
case "$1" in
```

```
    "verify")
```

```
        MODE="dry-run"
```

```
        echo -e "${YELLOW}===== ${NC}"
```

```
        echo -e "${GREEN}[ ] ( ) ${NC}"
```

```
        echo -e "${YELLOW}===== ${NC}"
```

```
        ;;
```

```
    "rename")
```

```
        MODE="real"
```

```
        echo -e "${YELLOW}===== ${NC}"
```

```
        echo -e "${RED}[ ] ! ${NC}"
```

```
        echo -e "${YELLOW}===== ${NC}"
```

```
        ;;
```

```
    "-h"|"--help")
```

```
        show_help
```

```
        exit 0
```

```
        ;;
```

```
    *)
```

```
        echo -e "${RED}[] : $1 ${NC}"
```

```
        show_help
```

```
        exit 1
```

```
        ;;
```

```
esac
```

```
#
```

```
font_files=$(find "${FONTDIR}" -type f \( -name "*.ttf" -o -name "*.otf" -o
```

```
-name "*.ttc" -o -name "*.woff" -o -name "*.woff2" \) 2>/dev/null)
```

```
if [ -z "$font_files" ]; then
```

```
    echo -e "${RED}[] '${FONTDIR}' . ${NC}"
```

```
    exit 0
```

```
fi
```

```
processed=0
```

```

renamed=0

IFS=$'\n'
for original_filepath in $font_files; do
    ((processed++))
    echo "-----"
    echo -e "${YELLOW}[*]          : $original_filepath${NC}"

    # 1. fc-scan
    fullname_string=$(fc-scan -f "%{fullname}" "$original_filepath"
2>/dev/null)
    echo -e "  ${PURPLE}[DEBUG] fc-scan raw value:${NC} '$fullname_string'"

    if [ -z "$fullname_string" ]; then
        echo -e "  ${RED}[] fullname          가          .          .${NC}"
        continue
    fi

    # 2.
    IFS=', ' read -ra tokens <<< "$fullname_string"
    selected_name=""
    first_token=""

    for i in "${!tokens[@]}; do
        token="${tokens[$i]}"
        token=$(echo "$token" | sed -e 's/^[[:space:]]*//' -e
's/[[:space:]]*$//')
        [ -z "$token" ] && continue
        [ -z "$first_token" ] && first_token="$token"

        if echo "$token" | perl -CS -ne 'exit 1 unless /[\\x{AC00}-
\\x{D7A3}\\x{3131}-\\x{318E}]/;' ; then
            [ -z "$selected_name" ] && selected_name="$token"
            echo -e "  ${BLUE}[DEBUG] Token[$i]:${NC} '$token' ${GREEN}(
)${NC}"
        else
            echo -e "  ${BLUE}[DEBUG] Token[$i]:${NC} '$token' (    /    )"
        fi
    done

    # 3.
    new_name_base="${selected_name:-$first_token}"
    sanitized=$(echo "$new_name_base" | sed 's/[[:space:]]/_/g' | sed
's/[\\/:\\*\\? "<>| ]//g' | sed 's/___*/_/g')

    # 4. 32          VF
    byte_size=$(printf "%s" "$sanitized" | wc -c)
    if [ "$byte_size" -gt 32 ]; then
        first_word="${new_name_base%% *}"
        sanitized_short=$(echo "$first_word" | sed 's/[[:space:]]/_/g' | sed
's/[\\/:\\*\\? "<>| ]//g')

```

```
        if [[ "$sanitized_short" =~ [vV][fF]$ ]]; then
            sanitized="$sanitized_short"
            echo -e "    ${YELLOW}[!]                VF          ->      :${NC}"
'$sanitized'"
        else
            sanitized="${sanitized_short}VF"
            echo -e "    ${YELLOW}[!]                -> VF   가:${NC} '$sanitized'"
        fi
    fi

    if [ -z "$sanitized" ]; then continue; fi

# 5.
    dir_path=$(dirname "$original_filepath")
    extension="${original_filepath##*.*}"
    final_new_filepath="$dir_path/${sanitized}.${extension}"

    counter=1
    while [ -e "$final_new_filepath" ] && [ "$final_new_filepath" !=
"$original_filepath" ]; do
final_new_filepath="$dir_path/${sanitized}(${counter}).${extension}"
        ((counter++))
    done

# 6.
    if [ "$original_filepath" = "$final_new_filepath" ]; then
        echo "    [*]                (                )"
    else
        echo -e "    ${GREEN}[+]                :${NC}"
        echo "        FROM: $(basename "$original_filepath")"
        echo "        TO:   $(basename "$final_new_filepath")"
        if [[ "$MODE" == "real" ]]; then
            mv "$original_filepath" "$final_new_filepath"
            echo -e "    ${GREEN}[✓]                ${NC}"
        fi
        ((renamed++))
    fi
done

echo "-----"
echo -e "    ! (    $processed    $renamed    )"

if [[ "$MODE" == "dry-run" && $renamed -gt 0 ]]; then
    echo ""
    echo -e "${YELLOW}                                :${NC}"
    echo -e "${GREEN}${BOLD}$0 rename${NC}"
fi
```

From:  
<https://atl.kr/dokuwiki/> - AllThatLinux!

Permanent link:  
[https://atl.kr/dokuwiki/doku.php/%ED%8F%B0%ED%8A%B8\\_%EC%9D%B4%EB%A6%84%EC%9C%BC%EB%A1%9C\\_%ED%8C%8C%EC%9D%BC%EB%AA%85\\_%EB%B3%80%EA%B2%BD](https://atl.kr/dokuwiki/doku.php/%ED%8F%B0%ED%8A%B8_%EC%9D%B4%EB%A6%84%EC%9C%BC%EB%A1%9C_%ED%8C%8C%EC%9D%BC%EB%AA%85_%EB%B3%80%EA%B2%BD)

Last update: 2025/12/24 01:19

